NASA GPU Hackathon 2022 Summary

Haoqiang Jin and Gabriele Jost

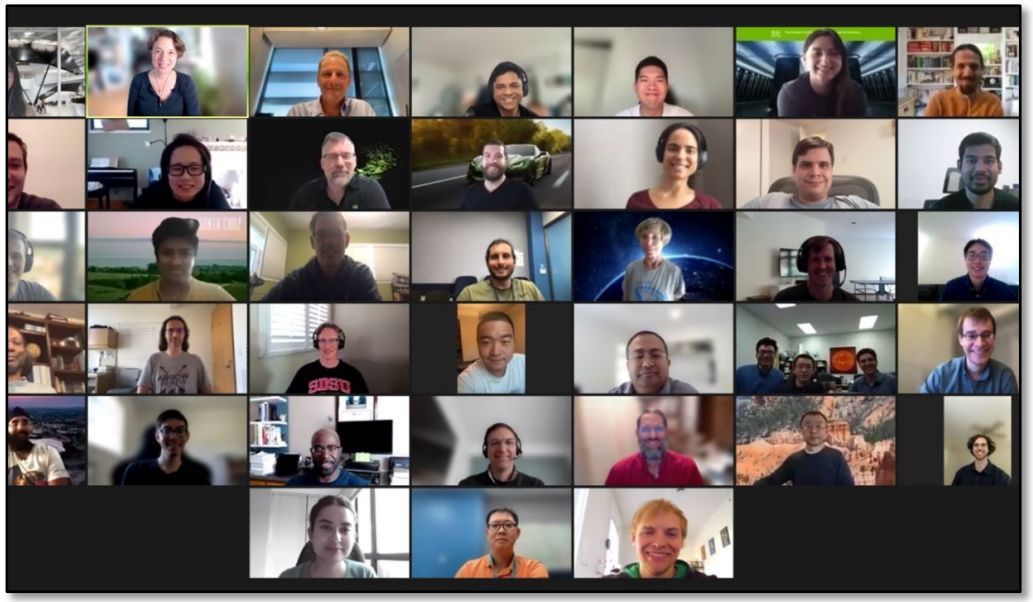
www.nas.nasa.gov/hackathon

September 19 and 26-28, 2022

HECC Hosts the Successful 2022 NASA GPU Hackathon

- HECC at NASA Ames and NVIDIA jointly organized the 2022 NASA GPU Hackathon to bring together application developers and computer experts to help get important applications from NASA centers and academic institutions running effectively on GPU nodes. The Hackathon was a virtual event and took place over four days on September 19 and 26-28.
  - 36 members representing 8 teams from three NASA centers (ARC, GSFC, JPL) and academic institutions participated in the hackathon. The teams were selected from 16 submissions through a review process.
  - The teams brought a variety of computationally intensive applications and accelerated their performance on GPUs using OpenACC, CUDA, CuPy, CuML, and other techniques.
  - The participants used Zoom and Slack for collaboration. All the teams met together in a daily "scrum" session to present their progress.
  - HECC and NVIDIA each provided nodes containing NVIDIA V100 and A100 GPUs for the teams to use.
- By the end of the Hackathon, most teams had made good progress and achieved considerable performance improvements on both GPUs and CPUs:
  - A key to the success was the pairing of teams with 18 expert mentors from NVIDIA, LBNL and NASA.
  - With optimization techniques and performance tools, teams were able to achieve application performance improvement ranging from 18% up to 100X compared to original codes prior to the hackathon.
  - All teams expressed appreciation of the opportunity and desire to continue working on their GPU projects after the hackathon.
- HECC will coordinate with NASA Langley to host next year's GPU hackathon.

IMPACT: The NASA GPU Hackathon allowed the teams to gain experience porting and optimizing their codes for GPUs increasing the performance of most their applications.



GPU Hackathon group photo showing 38 of the 66 participants and supporting staff. *Image courtesy of NVIDIA*

Selected Teams and Application Areas

Team Name	NASA Center or Institution	Application Domain	Algorithmic Motifs	Programming Approach	Achieved (Comparing with Pre-Hackathon)
IceSheet Modeling	Univ. of North Dakota	Regional-scale real glacier models	Iterative and matrix-free pseudo-transient algorithm	HPC / CUDA C	27% gain on single GPU
Mesirov Lab	UCSD	Analysis of genomic data	Non-negative matrix factorization	Rapids.ai, CuPy, MPI+OpenACC	18-40% gain
NU-HEXA	Northwestern Univ.	Four applications on algorithm development	AM-CFD, SCA, Convolution FEM, Convolution Tensor Decomposition	OpenACC, cuFFT, JAX, MATLAB	No gain for two apps; 100X for FEM; 45X for TD
LAVA Mini-App	NASA Ames	Computational Fluid Dynamics	Gradients and second order convective scheme	CUDA C++	298 MUPS on GPU (vs 50 MUPS on CPU)
UVA Bootes	Univ. of Virginia	Hydrodynamic and dust simulations in planet formation	Hydrodynamic and dust dynamics	OpenACC, CUDA, C++	8X GPU compared to 20-core CPU
Team JPL	JPL	Smoke plume and active fire identification	ML models, unsupervised RBMs and BIRCH clustering	PyTorch, CuPy / CuML	37X for image identifying; 4X per epoch for training
PICASO	NASA Ames	Modeling atmospheric properties	Thermal, transmission, reflected	CuPy, Numba.cuda,	30-80X speedup
Team Cauchy	UCLA, NASA Goddard	Multivariate Cauchy estimator for linear and non-linear dynamic systems	Kalman filter	CUDA-C/C++, LAPACK	5% improvement for a key kernel; identified an algorithm change that could result in >40% improvement

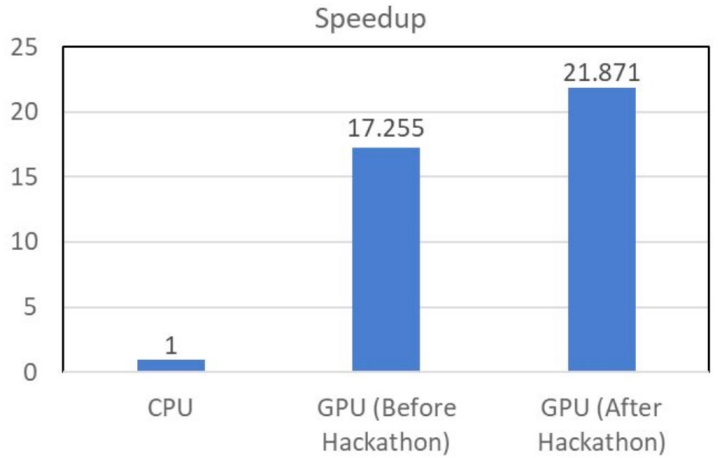
Team Accomplishments

IceSheet Modeling

Univ. of North Dakota

Application Background

- Leveraging GPU computing power to diminish spatial resolution constraints and improve computing speed for large-scale ice-sheet flow simulations to better predict sea level rise in a changing climate
- Unstructured meshes/finite elements, matrix-free solver



Speedup achieved on a single GPU

Hackathon Objectives and Approach

- Programming model – CUDA C
- Identify performance bottlenecks using nsight systems and nsight compute
- Refactor current CUDA C implementation to be more GPU-friendly

Technical Accomplishments and Impact

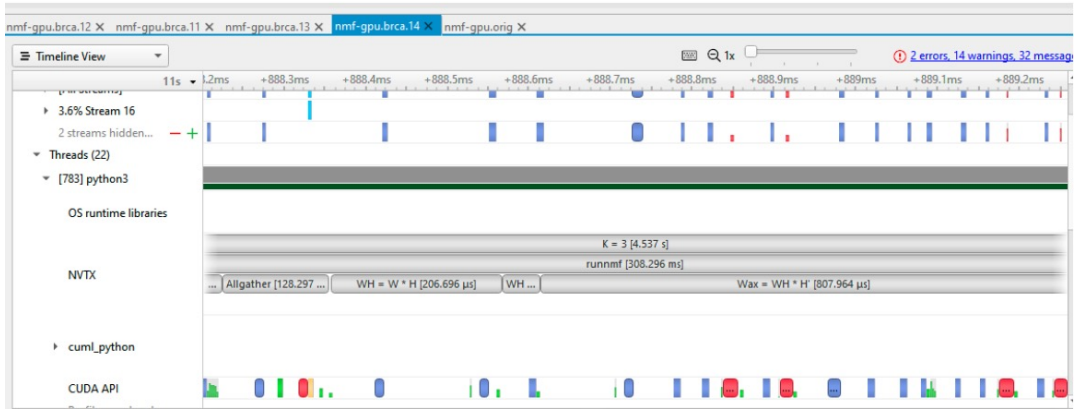
- Speedup compared to CPU implementation
  - Initial GPU implementation: ~ 17x
  - Revised GPU implementation: ~ 22x
- Reduced occurrences of thread idling/used registers or local variables

Mesirov Lab

Univ. of California San Diego

Application Background

- Non-negative Matrix Factorization (NMF) decomposes an array into smaller matrices that contain interpretable components.
- In the biological domain, a matrix of gene activity across samples can be decomposed as a superposition of cellular processes, regulatory pathways, and other biologically identifiable structures.
- NMF is highly iterative, compute-intensive, and relies heavily on matrix operations, and therefore scientists often use less intensive but less effective algorithms



Profiler output illustrating optimized execution of NMF method and calculation of cluster fit

Hackathon Objectives and Approach

- Move computation of cophenetic correlation coefficient (CCC) measure of cluster fit from host CPU into GPU
- Implement ability to distribute large datasets across multiple GPU cores and nodes
- Ensure that the implementation is portable to SDSC Expanse GPU nodes

Technical Accomplishments and Impact

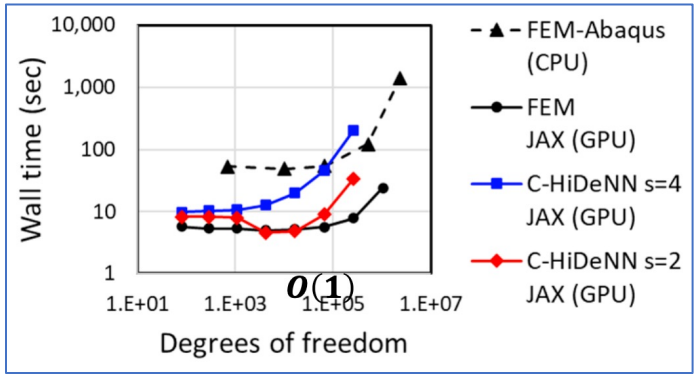
- Removed unnecessary transfers between host and GPU and implemented multi-threading
- Integrated rapids.ai memory manager to remove memory inefficiencies
- Observed 18% to 40% performance increase on test datasets
- We will deploy this on our public gateway, allowing the worldwide genomics community to utilize this functionality

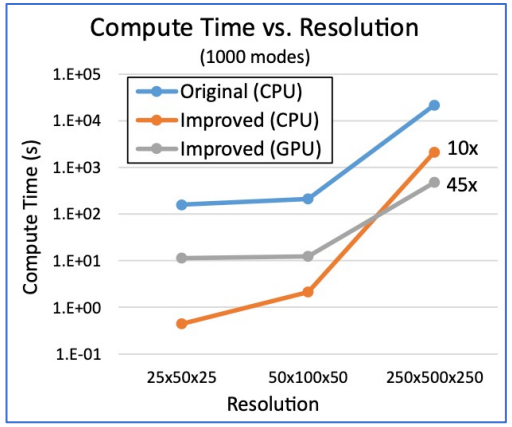
NU-HEXA

Northwestern Univ.

Application Background

- AM-CFD: A thermal CFD solver for additive manufacturing process that can tackle large scale problems.
- SCA: Clustering based multiscale analysis
- Conv-FEM: A new interpolation technique that gives orders of magnitude better accuracy and convergence
- C-TD-TO: Convolution tensor decomposition for topology optimization





Speedup achieved for Conv-FEM (left) and the HOPGD module in C-TD-TO (right)

Hackathon Objectives and Approach

- AM-CFD: Matrix inversion/solver acceleration using OpenACC
- SCA: cudaFFT for offline computation, LU inverse acceleration
- Conv-FEM: Parallelizing the algorithm to solve problems with larger degrees of freedom – use JAX built-in solvers
- C-TD-TO: Modify the HOPGD module to accelerate the decomposition procedure – vectorize operations

Technical Accomplishments and Impact

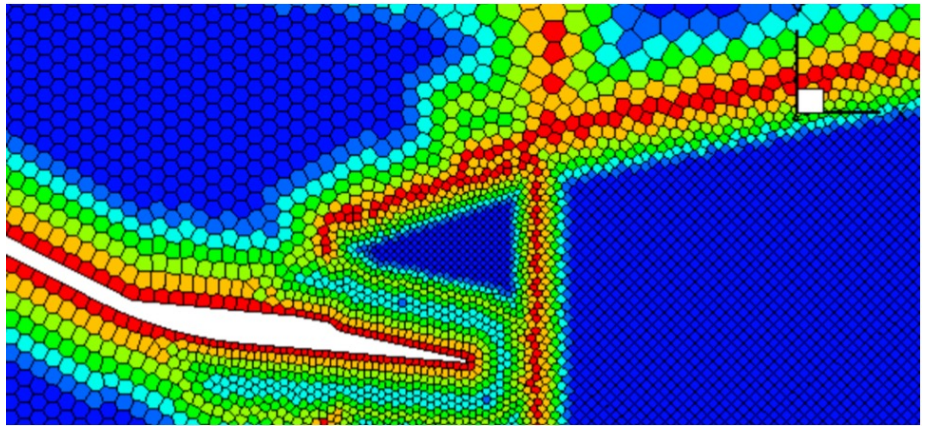
- No speedup for AM-CFD and SCA
  - Still learning programming approaches
- Conv-FEM
  - Overall 100x speedup with JAX parallelization
- C-TD-TO
  - 10x speedup on CPU and 45x speedup on GPU

LAVA Mini-App

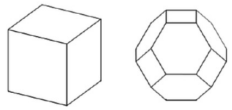
NASA Ames

Application Background

- Arbitrary polyhedral unstructured CFD solver.
- Targeting large-scale, wall-modeled LES simulations O(1e9) elements in the full application.
  - CPU performance is already highly optimized
  - Interested in finishing GPU porting of the basic scheme with CUDA
  - Language: C++ and CUDA



Polyhedral and hexahedral meshes
Hex: 6 faces/cell, Polyhedral: 14 faces/cell



Hackathon Objectives and Approach

- Profile the current implementation and address bottlenecks
- Implement gradient computation
- Achieve higher performance compared to CPU
- Resolve existing issues with NVIDIA compilers

Technical Accomplishments and Impact

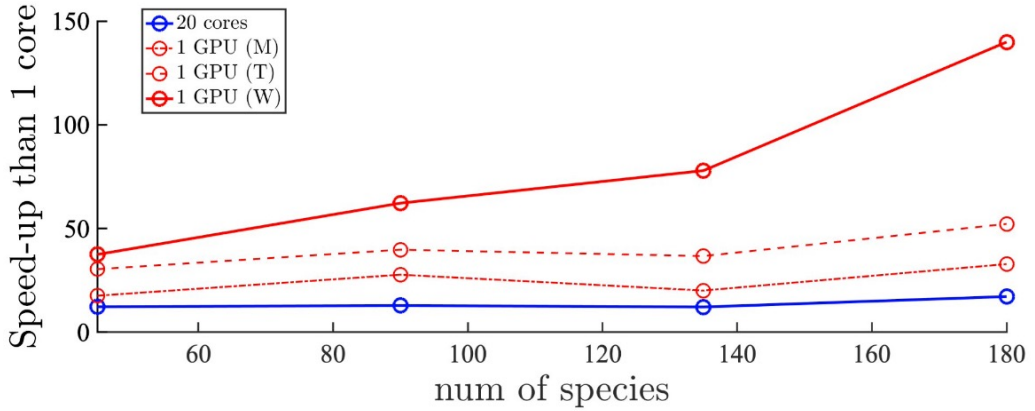
- Implemented gradients and second order convective scheme
- Utilizing ncu, optimized register & shared memory usage to increase occupancy
- Performance achieved on a single GPU
  - Hex mesh 128^3: 298 MUPS (6X compared to 40-core CPU)
  - Poly mesh 128^3: 94 MUPS
- Future work: Gradient computation, viscous fluxes

UVA Bootes

Univ. of Virginia

Application Background

- Hydrodynamics + Dust dynamics and grain growth simulation.
- Studies planet formation – How Earth was built from dust grains smaller than sand?



Performance improvement with progress in day 1 (M), day 2 (T) and day 3 (W)

Hackathon Objectives and Approach

- OpenACC
- Simulation with GPU and output with CPU
- Focus on thread optimization and data movement

Technical Accomplishments and Impact

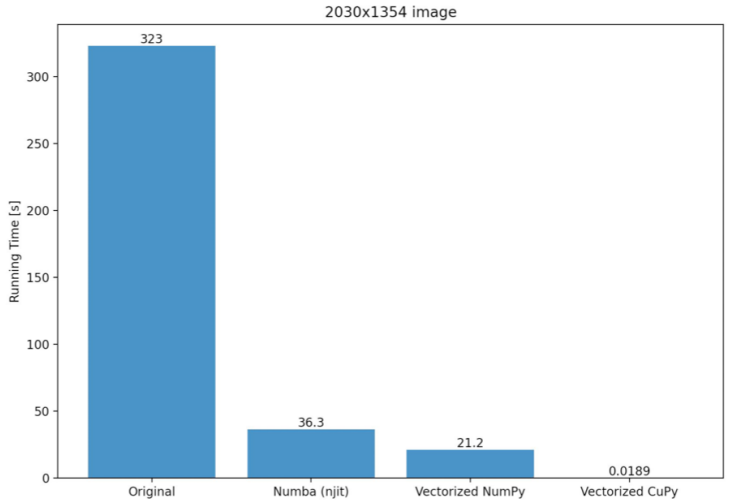
- New GPU-accelerated hydrodynamics code
  - 140x faster comparing to single CPU
- Impact statements
  - Understanding of Planet formation
  - Finding aliens?

Team JPL

Jet Propulsion Laboratory

Application Background

- ML models for smoke plume and active fire identification.
- Methodology to make data more “plug-and-play”.
- Input data: images from multiple airborne and satellite instruments during NASA’s field campaign in 2019.
- Unsupervised ML approach to the problem of detecting and tracking wildfire smoke plumes through sequences of L1 images acquired by multiple remote sensing instruments.



Running time improvement from different approaches for one large block of nested loops

Hackathon Objectives and Approach

- Implementation any necessary GPU-interface improvements/optimizations with PyTorch, CuPy/CuML
- Integration of deeper architecture that uses convolutional layers for segmentation
- Clustering code uses scikit-learn

Technical Accomplishments and Impact

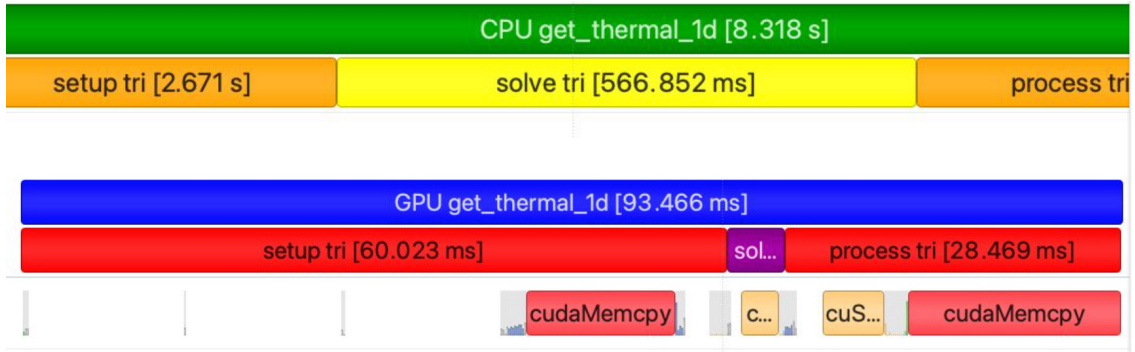
- Preprocessing bottleneck was eliminated by vectorizing large loops using CuPy arrays
  - Speed up running times for loading data by ~37x
- Improvements to DDP implementation
  - 4x speedup per epoch during training
- Simplification of architecture swapping

PICASO

NASA Ames / BAERI

Application Background

- Interpreting atmospheric properties of exoplanet and brown-dwarf atmospheres from observational data.
- This might include inferring the presence of gases like CO_2, CH_4, H_2O, etc in these far away worlds.
- This requires running models for ~million times.
- With CPUs this can take several days, speeding up these atmospheric models is critical to support JWST and other NASA missions.



GPU enabled code speed ups of up to 80x compared to single CPU

Hackathon Objectives and Approach

- Python language with SQL data structure
- Radiative transfer functions in python
- numba, cupy
- NSIGHT profiling

Technical Accomplishments and Impact

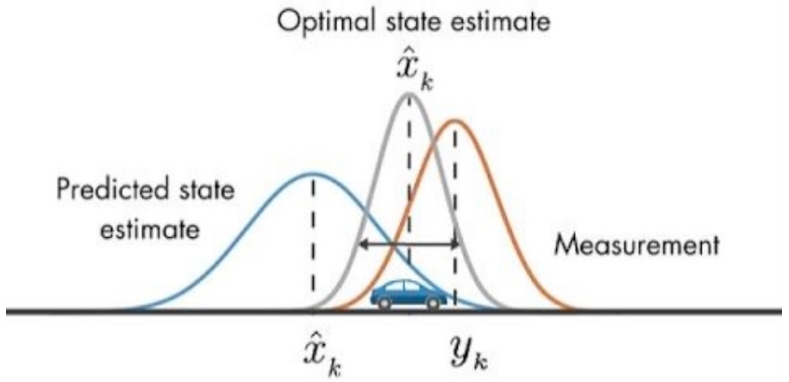
- Learn GPU & profiling basics, transfer critical functions from Numba CPU to Numba cuda
- Using a hybrid of cupy and numba.cuda
- Speedups of 30-80x compared to CPU
- Enables rapid characterization of exoplanet atmospheres with JWST and other space and ground telescopes

Team Cauchy

UCLA / NASA Goddard

Application Background

- Multivariate state estimator based on Cauchy probability density function for linear and nonlinear dynamic systems.
- Good for stochastic environments with volatile and impulsive noises (deep space, low-earth orbit, missile interception, stock market).
- Distributed computation of the mathematics allows for real-time performance.



Multivariate Cauchy estimator uses Gaussian probability density functions to model uncertainty

Hackathon Objectives and Approach

- CUDA-C/C++
- Intel MKL, LAPACK, CUB
- NSight Systems + Compute

Technical Accomplishments and Impact

- Identified hot-spot areas in kernel taking the longest
- Maxed out compute in kernel
  - Achieved 5% performance improvement
- Identified an important algorithmic improvement that could result in 50% performance improvement